Chemical Engineering: 3E4
Process model formulation and solution
McMaster University: Take-home examination

Take-home exam: 5 days
17 to 22 November 2010

Instructor: Kevin Dunn
dunnkg@mcmaster.ca

Instructions

The purpose of this exam is only a little different to other exams. The difference is that this exam will assess your ability to answer more realistic problems that require more time, thought, and access to a computer. You are expected to use **any appropriate tools** to solve the problems in this exam, particularly the tools learnt in this course.

The 3E4 course teaches you a collection of tools that will be useful in your engineering career. And the challenges you will face do not come neatly posed as problems of the type: "Solve this linear system using Gauss elimination" – rather you need to reformulate your challenge and pick the appropriate tool to solve the problem, taking into account the level of complexity.

In that regard there are many correct ways to achieve a result. In this exam about 80% of the grade will be given for **how** you solve the problem, and we will pay particular attention to the **justifications** you provide along the way. You must clearly outline **why** you choose the method and **why** you did not choose alternative methods. The **accuracy of how you implement your method** is also important, but the actual solution is of little value for determining your grade.

Also, as you will find in your career as an engineer, you will have to apply tools that you learnt about long ago, tools just learnt recently, or you may have do some research and figure out how to use a tool you've never used before. This exam has those aspects as well.

Some other points are:

- Complete this exam with 1 other person, or by yourself.
- Please identify the person you work with at the top of your answer submission. **Please note: submit one solution per group**.
- Please attribute any sources of reference you used (textbooks, websites, and so on).
- The intention of the group work is that you discuss the questions and collaborate in the same way you have done in the assignments.
- But **do not share** any electronic files (e.g. Word documents, source code) outside your group; take care in the computer labs to safeguard your work.
- Like any other exam, neither the TAs nor myself are able to answer direct questions about the exam. Similarly you *should not* look for help about a *specific question* from other resources (e.g. asking for help with the question on a website, friends, other students in the class, *etc*).
- You may use the course notes and any other textbooks and resources though.
- There is no make-up for this exam.
- You will benefit from going through the software tutorials on the course website.
- Your answers should preferably be typed up.
- Your answers must be structured like a report and flow together logically. Please do not submit pages and pages of computer printouts and source code in appendices - unfortunately this will be penalized.
- Hand out date: 17 November 2010,

- **Due date:**
    - Paper hand-in: by **17:30 on 22 November 2010, in class**
    - Totally electronic hand in: email to dunnkg@mcmaster.ca before the above time and date.
    - A valid electronic hand in must be in PDF format *only* (no separate Word, MATLAB or Python files). If you are on Windows, you could use PDFCreator to make PDF's; Macs and Linux have built-in PDF capability.

Note: There are a maximum of 30 grade points available in the 5 questions, plus a variable amount of bonus points.

Only brief solutions are provided as we covered all the questions in class. The grading schedule is provided with each question.

Question 1 [7 = 2 + 2 + 2 + 1]

Background: Chemical engineers have made many contributions to the medical field; similarly in this question you must use your engineering knowledge in a different context.

Consider the case of injecting a medical treatment into a person's blood system. Subsequent to the injection, the body starts to metabolize (break down, or use up) the drug. In this study a drug was injected into the patient, and samples taken over the next 10 hours. This table reports the concentration of the drug in the blood stream, C:

t [hour]	1	2	3	5	6	8	10
C [mg/L]	10.1	7.5	5.7	3.1	2.4	1.3	0.7

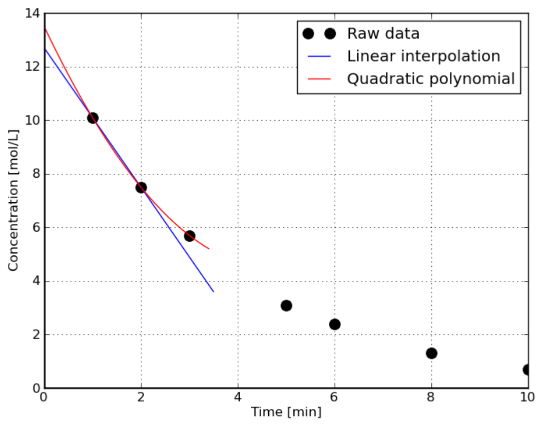
Medical models often use a well-mixed reactor to approximate the time-dependent behaviour of parts of the body: e.g. kidneys, stomach, and other organs. In this question you can assume the blood system to be a well-mixed CSTR, operating in batch mode: i.e. no inlet, or outlet, and of constant volume (about 5L for a typical adult). Assume the injection immediately mixes with all the blood, and shows up as $C(t = 0)$.

1. Provide an estimate of the drug's concentration at $t = 0$ hours. Please justify your choice of method, and provide a plot to assess if your estimate is reasonable.
2. Please provide the lowest error estimate of $\frac{dC}{dt}$ for every data point in the table (without using Richardson's extrapolation).
3. Next assume the drug is eliminated from the patient's blood into the rest of the body using a first-order rate expression, $r = -kC$. Obviously the mechanism by which the drug is metabolized is much more complex, but we don't know the cellular pathways and their rate constants: so lump them together and assume the overall mechanism is first order. Derive the dynamic equation that would describe the concentration of the drug in the bloodstream as a function of time.
4. Using your values from part 2, calculate the average rate constant. **Bonus mark of 0.5** if you use least squares regression instead of just calculating the average.

Solution

1. The time range given in the table does not contain $t = 0$. So, any method used to estimate $C(t = 0)$ will be an extrapolation. A simple way to estimate $C(t = 0)$ is by a linear extrapolation, which can be justified by the small distance between $t = 0$ and $t = 1$ and also $t = 1$ and $t = 2$ (so a linear behaviour can be assumed over that range). We pass a line through $(1, 10.1)$ and $(2, 7.5)$ as $C = -2.6t + 12.7$, so $C(t = 0) = 12.7$.

   A spline or polynomial fit, through the first 3 points or so would also be a valid, though longer approach. The polynomial through the first 3 points is $C(t) = 13.5 - 3.8t + 0.4t^2$, so here $C(t = 0) = 13.5$. Both fits are illustrated in the figure.



As explained in class, neither value is correct in practice, since $C(t = 0) = 0$ the moment the injection is administered.

2. The lowest error estimate of $\frac{dC}{dt}$:

t	C	Lowest error method	*Result*
1	10.1	forward diff	-2.60
2	7.5	central diff	-2.20
3	5.7	central diff	-1.750
5	3.1	central diff	-1.033
6	2.4	backward diff	-0.70
8	1.3	central diff	-0.4250
10	0.7	backward diff	-0.30

3. In this problem our *control volume* to be modelled is the entire human's body. So, the bloodstream inside the body makes up a batch system because there is no blood flow into/out of the body. Another assumption is that we have well-mixed system; that is there is no spatial distribution of quantities through the body. This assumption means that the blood circulation is fast enough, making a well-mixed environment. Also, the system is aqueous and the volume is constant. Given these assumptions, the dynamical model would be as follows:

$$\text{Accumulation} = \text{Input} - \text{Output} + \text{Generation} - \text{Consumption}$$
$$\frac{d(VC)}{dt} = 0 - 0 + 0 - rV$$
$$\frac{dC}{dt} = -kC$$

4. In the model we just obtained, the parameter k is not known. But we can compute an approximation of it using the dynamical model, and the estimated values of the derivative $\frac{dC}{dt}$ at different concentrations C. One way is to obtain values of k_i corresponding to each data point C_i, then computing an average of the k_i values:

t	C	$\frac{dC}{dt}$	k
1	10.1	-2.60	0.2574
2	7.5	-2.20	0.2933
3	5.7	-1.750	0.3070
5	3.1	-1.033	0.3332
6	2.4	-0.70	0.2917
8	1.3	-0.4250	0.3269
10	0.7	-0.30	0.4286

The average rate constant is $\bar{k} = \dfrac{\sum_{i=1}^{7} k_i}{7} = 0.3197$

Now we try to obtain k using regression. Let us denote $\frac{dC}{dt}$ by y, and $y_{\text{model}} = \frac{dC}{dt} = k(-C)$ represent the derivative that we obtain from the model. On the other hand, we also calculated the derivatives in Part 1 using the experimental data (even though approximate!). Let us denote them by $y_{\text{exp}} = (\frac{dC}{dt})_{\text{exp}}$. Now we can set up a least square regression as follows.

$$\min \quad J = \sum_{i=1}^{7} (y_{\text{model,i}} - y_{\text{exp,i}})^2$$

In order to minimize the above error function J, we have to equate its derivative w.r.t. k to zero. That is, $\frac{dJ}{dk} = 0$

$$\frac{dJ}{dk} = 2\sum_{i=1}^{7} (\frac{dy_{\text{model,i}}}{dk})(y_{\text{model,i}} - y_{\text{exp,i}}) = 0$$

Note that $y_{\text{model,i}} = k(-C_i)$. So $\frac{dy_{\text{model,i}}}{dk} = -C_i$. Substituting these into the above equation will give:

$$\frac{dJ}{dk} = \sum_{i=1}^{7} (C_i)(k(-C_i) - y_{\text{exp,i}}) = 0$$

Rearranging the above equation and solving it for k (left to you as an exercise) will complete the derivation:

$$k = \frac{\sum_{i=1}^{7} -C_i y_{\text{exp,i}}}{\sum_{i=1}^{7} C_i^2}$$
$$k = \frac{-(10.10 \times -2.60 + 7.50 \times -2.20 + 5.70 \times -1.75 + 3.10 \times -1.033 + 2.40 \times -0.70 + 1.30 \times -0.425 + 0.70 \times -0.30)}{10.10^2 + 7.50^2 + 5.70^2 + 3.10^2 + 2.40^2 + 1.30^2 + 0.70^2}$$
$$k = 0.2803$$

Note: According to the model $y_{\text{model}} = \frac{dC}{dt} = k(-C)$, the slope k is the only coefficient to be obtained. But if you use MATLAB's command `polyfit` or Excel's linear regression, by default they will calculate two coefficients, one is the slope and the other one is the intercept. So be careful: having an intercept may not be physically meaningful in your system. In such a case (as in this problem), you should have MATLAB or Excel set the intercept to zero or do the regression yourself.

Note: we showed an alternative method in class that first integrated the equation to obtain $\ln[C] = \ln[C(t = 0)] - kt$. By regressing values of $\ln[C]$ onto t we can estimate both the rate constant and the initial concentration. Using this method you will obtain $k = 0.295$ hours^{-1} and $C(t = 0) = 13.7$ mg/L.

Question 2 [5 = 1 + 1 + 3]

A batch bioreactor (well-mixed tank, but no inlet or outlet) is charged with substrate at time $t = 0$ to start a reaction that will consume the substrate. The reaction mechanism is too complex to model exactly, but previous experience suggests it is roughly a first-order reaction. The apparent first-order rate expression is $r = kS$, where $k = 0.58$ hours^{-1}, and S is the concentration of the substrate being depleted.

1. Show that the dynamic rate of change of S is given by:

$$\frac{d\mathsf{S}(t)}{dt} = -k\mathsf{S}(t)$$

Also state any assumptions that were made in arriving at this expression.

2. The time required to consume the substrate from a starting level of 260 g/L to 5 g/L can be calculated by analytical integration of the above expression. What is the total time required?

3. Had you forgotten how to integrate the expression analytically you could resort to using the trapezoidal rule or Simpson's 1/3 rule. **Without** implementing either method, calculate how many panels would be required to ensure a numerical estimate of the time to achieve the above conversion is accurate within 0.01 hours. In other words, how many panels are required to solve the left hand side integral

$$\int_{260}^{5} -\frac{1}{k\mathsf{S}}\, d\mathsf{S} = \int_{0}^{t} dt = t$$

to that level of accuracy:

   - when using the trapezoidal rule;
   - when using Simpson's 1/3 method.

Solution

1. Ordinary batch system balance, with no in or out flows, only a reaction term $-k\mathsf{S}$ and the accumulation term, $\frac{d\mathsf{S}}{dt}$. The key assumption is that all other properties, such as batch volume and physical properties are constant. The system is also assumed to be well-mixed, as stated. Other assumptions would be that the energy inputs and output are negligible (i.e. shaft work, heat losses to the environment and specifically that the reaction rate is not a function of temperature).

2. $\ln(\frac{5}{260}) = -0.58t$ = 6.81 hours, or 409 minutes.

3. This question turned out to be less useful than I had intended. I had intended that you calculate and realize how a much smaller number of panels are required for Simpson's rule than the trapezoidal rule - in order to get the same level of error. However, due to:

   - an unfortunate error in the PDF course notes
   - an integral that wasn't expressed in terms of time
   - a difference between the 5th and 6th edition of the course textbooks

   I've decided to omit this question from the exam, but give 1 bonus point to everyone that seriously attempted the question (i.e. the exam is out of 27 points, but you probably got a bonus point).

   The way to attempt this question was to first rearrange the left-hand integral so that it had units of time, since the error was asked to be below 0.01 hours. The original question was expressed as shown below in the first line, but then later changed to read as the second line.

$$\int_{260}^{5} \frac{1}{\mathsf{S}}\, d\mathsf{S} = -k\int_{0}^{t} dt = -kt$$
$$\int_{260}^{5} -\frac{1}{k\mathsf{S}}\, d\mathsf{S} = \int_{0}^{t} dt = t$$

   We can use the form shown in the first line, as long as we multiply our answer by k afterwards.

   So the function we are integrating is $f(S) = S^{-1}$ and has units of L/g. We will require derivatives of this function, $f''(S) = 2S^{-3}$, with units of L^3/g^3, and $f^{(4)}(S) = 24S^{-5}$ with units of L^5/g^5.

   The main problem was from the PDF course notes. There was an inaccuracy that said you could find the number of panels for the trapezoidal rule if this expression was satisfied:

$$\frac{(b-a)^3}{12n^2} f''(\xi) < 0.01 \qquad \text{for some value of } \xi \text{ in the range } [5, 260]$$

   However, the correct expression for the error is:

$$\left| -\frac{(b-a)^3}{12n^3} \sum_{1}^{n} f''(\xi_i) \right| < |-kt| = (0.58)(0.01) = 0.0058$$

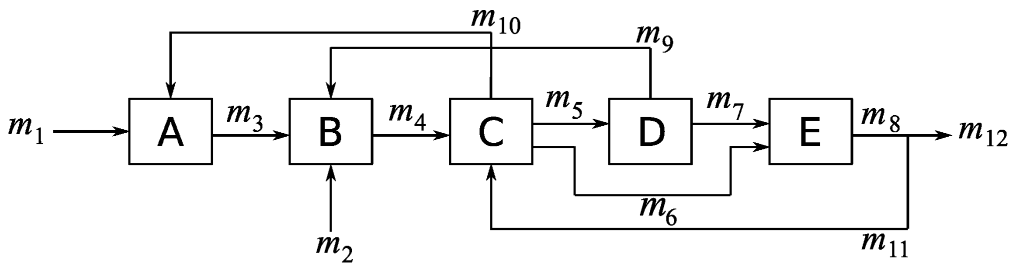
Note the units: a and b are in g/L while $f''(\mathrm{S})$ has units of $\mathrm{L}^3/\mathrm{g}^3$, so everything balances.

The problem is evaluating the summation: you have to find the worst value of $f''(\mathrm{S})$ within every panel, but the purpose of this question is to find the number of panels! So the course textbook shows that we rather find the average error, and that was the equation in the PDF course notes, except it was incorrect to have the part that ξ had to be in the range from $[a, b]$.

When I set the exam I used the PDF course notes, not the course textbook, so I was unaware of this complication in the question. My apologies if you spent a lot of time on this question.

Question 3 [7 = 3 + 1 + 3]

The following question is based on the flowsheet below, where the m_i values are the mass flow of the species of interest, the reactant. The purpose of the flowsheet is to eliminate the incoming species at $m_1 = 10$ kg/hour and convert it to by-products. There is some residual reactant leaving the flowsheet in stream m_{12}.



The following additional information is known:

- $m_1 = 10$ kg/hour.
- $m_2 = 2$ kg/hour.
- Unit A operates with 40% efficiency; i.e. of all the material coming in, only 40% of it is converted to other products.
- Unit B is just a mixing tank.
- Unit C is just a mixer/splitter and is operated so that the mass-flow in the outlets are all in the same proportion.
- Unit D operates with 82% efficiency, so 18% of the incoming stream is left unconverted. The two outlets are split in equal proportion as well.
- Unit E operates with 15% efficiency to convert the incoming streams to by-products.
- The split after unit E is operated such that αm_8 is recycled in stream 11, while $(1 - \alpha)m_8$ exits in stream m_{12}.

1. Write a system of equations in the form $\mathbf{A}\mathbf{x} = \mathbf{b}$ that can be used to calculate the mass flow rate of every stream in the flowsheet.

   The row order in a system of equations has no effect on the solution $\mathbf{x}$. But to make grading this question easier, please write your system of equations using $\mathbf{x} = [m_1, m_2 \ldots, m_{11}, m_{12}]$ and so that matrix $\mathbf{A}$ has 1's on the diagonal.

2. Solve this system of equations using computer software, at the value of $\alpha = 0.20$.
   - Show your matrix $\mathbf{A}$
   - Report the flow of the species in every stream, and in particular, the flow in stream m_{12}.

3. The next objective is to adjust the recycle ratio, α, so that the mass flow of the species leaving the flowsheet is $m_{12} = 1.0$ kg/hour.
   - You may use any technique you like to find α, but every time you solve the system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$ for a given value of α it will count as one function evaluation.

- If you plan to plot the system at various values of α, then each point on the plot is counted as a function evaluation.
- The solution in part 2, with $\alpha = 0.20$ counts as one function evaluation already, and should be your starting point for solving this part of the question.
- Clearly explain your strategy.
- Report the total number of function evaluations you used to achieve an α value that is accurate within 1×10^{-2}. There is **one bonus mark** available for getting an answer within 5 function evaluations or less.

Solution

1. *Marking scheme:*
   - 3/12 = 0.25 marks awarded for each equation in the system
   - Small errors in transferring equations to the $Ax = b$ form were ignored (where initial equation derivations were given)

   The first part of this question tasks us with developing the steady state mass balances for this system. Since we are dealing solely with the mass flows between units and not the internal dynamics of each unit operation this simplifies our modelling substantially and reduces the number of assumptions we need to make. We already know that mass must obviously be conserved, so really the only assumption we need to make is that the generalizations of each unit operation given above are correct (kind of an arbitrary assumption). Since we are operating at steady state we know that the overall mass balance around each unit must satisfy the following equality for the species of interest:

$$\{\text{Species in}\} - \{\text{Species out}\} + \{\text{Species Generated}\} - \{\text{Species consumed}\} = 0$$

Reactor A

Here we are given information on the overall mass conversion (40%) of reactor A and so know the consumption term of the reactor. We also know the inlet and outlet flows from the diagram itself.

$$\begin{aligned}\{m_1 + m_{10}\} - \{m_3\} + \{0\} - \{(0.40)\,(m_1 + m_{10})\} &= 0\\ m_3 &= (1 - 0.40)\,(m_1 + m_{10})\\ m_3 &= (0.60)\,(m_1 + m_{10})\end{aligned}$$

Mixing tank B

$$\begin{aligned}\{m_2 + m_3 + m_9\} - \{m_4\} + \{0\} - \{0\} &= 0\\ m_4 &= m_2 + m_3 + m_9\end{aligned}$$

Mixing tank C

Here we are given additional information on the exact splits of the outlet terms (1/3 for each outlet) and so can partition the overall mass balance into 3 separate balances.

$$\begin{aligned}m_5 = m_6 = m_{10} &= \left(\frac{1}{3}\right)\{\text{species out}\}\\ m_5 &= \left(\frac{1}{3}\right)(m_4 + m_{11})\\ m_6 &= \left(\frac{1}{3}\right)(m_4 + m_{11})\\ m_{10} &= \left(\frac{1}{3}\right)(m_4 + m_{11})\end{aligned}$$

Reactor D

Here we are given information on the overall mass conversion (82%) of reactor D and so know the consumption term of the reactor. We are also given additional information on the exact splits of the outlet terms (1/2 for each outlet) and so can partition the overall mass balance into 2 separate balances.

$$m_7 = m_9 = \left(\frac{1}{2}\right)\{\text{species out}\}$$

$$\{\text{species out}\} = (0.18)\, m_5$$

$$\begin{aligned} m_7 &= \left(\frac{1}{2}\right)(0.18)\, m_5 \\ m_7 &= (0.09)\, m_5 \end{aligned}$$

$$\begin{aligned} m_9 &= \left(\frac{1}{2}\right)(0.18)\, m_5 \\ m_9 &= (0.09)\, m_5 \end{aligned}$$

Reactor E

Here we are given information on the overall mass conversion (15%) of reactor E and so know the consumption term of the reactor. We also know the inlet and outlet flows from the diagram itself.

$$\begin{aligned} m_8 &= (1 - 0.15)\,(m_6 + m_7) \\ m_8 &= (0.85)\,(m_6 + m_7) \end{aligned}$$

Final split

Here we are given information on the split following Reactor E as a function of split factor α.

$$m_{11} = \alpha\,\{\text{species out}\}$$

$$m_{12} = (1 - \alpha)\,\{\text{species out}\}$$

$$\{\text{species out}\} = m_8$$

$$\begin{aligned} m_{11} &= \alpha\,\{\text{species out}\} \\ m_{11} &= \alpha m_8 \end{aligned}$$

$$\begin{aligned} m_{12} &= (1 - \alpha)\,\{\text{species out}\} \\ m_{12} &= (1 - \alpha)\, m_8 \end{aligned}$$

Extra info

So far we have been able to develop 10 steady state balances. However, we have 12 variables (13 if you count α) and so we need 2 additional pieces of information and an α value to make this system of equations solvable.

Luckily these have been provided in the problem statement in the form of the specified inlet flows m_1 and m_2.

$$m_1 = 10.0\ [kg/hour]$$

$$m_2 = 2.0\ [kg/hour]$$

$$\alpha = 0.2$$

Converting the system to the form $Ax = b$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.6 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -0.6 & 0 & 0 \\ 0 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{3} & 1 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{3} & 0 \\ 0 & 0 & 0 & -\frac{1}{3} & 0 & 1 & 0 & 0 & 0 & 0 & -\frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & -0.09 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.85 & -0.85 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.09 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 1 & -\frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\alpha & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -(1-\alpha) & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \\ m_8 \\ m_9 \\ m_{10} \\ m_{11} \\ m_{12} \end{bmatrix} = \begin{bmatrix} 10 \\ 2 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

2. *Marking scheme:*

   - 0.75 for correct code or process used to solve the problem
   - 0.25 for correct answer

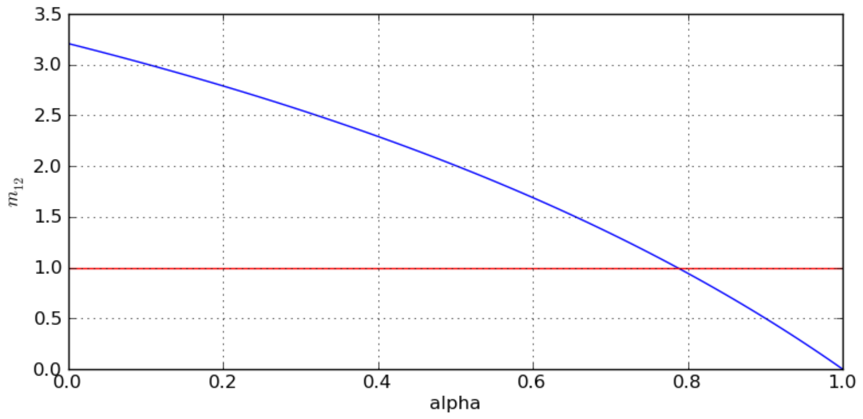
$$\begin{aligned} \mathbf{x} &= [m_1,\, m_2\, \ldots\, ,\, m_{11},\, m_{12}] \\ &= [10.0, 2.0, 8.26, 10.6, 3.77, 3.77, 0.34, 3.49, 0.34, 3.77, 0.7, \mathbf{2.79}] \end{aligned}$$

3. *Marking Scheme:*

   Given the open ended nature of this question it is hard to provide a hard and fast marking scheme. Roughly speaking 0.5 marks were awarded for acknowledging the type of problem and properly coding/explaining the NLAE function, 2.0 marks were awarded for properly explaining/coding/executing your chosen root finding method, and 0.5 marks were awarded for the final answer. Small mistakes were ignored (ex. typos) as long as not too many were encountered but serious flaws in logic were naturally penalized.

   The crux of solving this problem lies in realizing that if you treat the LAE in part 1 as a black box function of one variable (alpha) then this problem boils down to a root finding exercise involving one NLAE. Therefore you can apply any of the NLAE techniques (barring Newton-Raphson because you do not know the exact function, thus black box...) covered in slide set C. In order to solve the system in 5 - 6 iterations (and thus get the bonus marks) the best approach to apply would be the secant method. I found that whether the system solved in 5 or 6 iterations depended on the second point provided as an initial guess. The function itself has the basic shape shown below (note that this is the function $f(\alpha) = m_{12}(\alpha) - 1.0$). It is easy to see that the function has a very gentle slope and so is very well approximated by a straight line, which explains why the secant method works so well.

   Regardless of the method you end up using you must start by defining a function file to calculate $f(\alpha)$. Since we are trying to solve for when $m_{12} = 1.0$ it is also important to remember to subtract 1.0 from the m_{12} value calculated using the LAE system from part 1, so that we are dealing with an appropriate root finding function form. The secant method will be used here to provide an example of how to approach this problem, but really any root finding method would be acceptable as long as you justified your choice and explained your process.



Note that the MATLAB files have been appended to make the display look nicer in this document but in order to run the code you would need to split the provided code into its three original files as indicated.

MATLAB code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This bit of script should be saved as q3_p3_NLAE.m %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function m12_dif = q3_p3_NLAE(alpha)

A = [1,0,0,0,0,0,0,0,0,0,0,0;
         0,1,0,0,0,0,0,0,0,0,0,0;
         -0.6,0,1,0,0,0,0,0,0,-0.6,0,0;
         0,-1,-1,1,0,0,0,0,-1,0,0,0;
         0,0,0,-(1/3),1,0,0,0,0,0,-(1/3),0;
         0,0,0,-(1/3),0,1,0,0,0,0,-(1/3),0;
         0,0,0,0,-0.09,0,1,0,0,0,0,0;
         0,0,0,0,0,-0.85,-0.85,1,0,0,0,0;
         0,0,0,0,-0.09,0,0,0,1,0,0,0;
         0,0,0,-(1/3),0,0,0,0,0,1,-(1/3),0;
         0,0,0,0,0,0,0,-alpha,0,0,1,0;
         0,0,0,0,0,0,0,-(1-alpha),0,0,0,1];

b = [10, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]';
m = A\b

m12_dif = m(12) - 1.0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This bit of script should be saved as q3_p3_Secant.m %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function y = q3_p3_Secant(func,x0,x1,tol,maxit,relax)

% Initialization
count = 0;
fprintf('%10s\t%10s\t%12s\t%12s\t%12s\t%12s\n','ITERATION',...
        'F_COUNT','Xk','F(Xk)','Xk_tol','F_tol')

f0 = func(x0);
count = count + 1;
fprintf('%10d\t%10d\t%12.4d\t%12.4d\t%10s\t%10s\n',...
        -1,count,x0,f0,' ',' ')

f1 = func(x1);
count = count + 1;
```

```
fprintf('%10d\t%10d\t%12.4d\t%12.4d\t%10s\t%10s\n',...
        0,count,x1,f1,' ',' ')

x_dif_rel = (x1-x0)/(x1);
iter = 0;

while((abs(f1) > tol || abs(x_dif_rel) > tol) && iter < maxit)

  dfdx = (f1-f0)/(x1-x0);
  x0 = x1;
  f0 = f1;
  x1 = x1 - relax*f1/dfdx;
  f1 = func(x1);
  x_dif_rel = (x1-x0)/(x1);
  iter = iter + 1;
  count = count + 1;
  fprintf('%10d\t%10d\t%12.4d\t%12.4d\t%12.4d\t%12.4d\n',...
          iter,count,x1,f1,abs(x_dif_rel),abs(f1))
end

y = x1;
if(iter >= maxit)
    fprintf('\n\n Maximum number of iterations exceeded! \n\n')
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This bit of script should be saved as q3_p3_main.m %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tol = 0.01;
maxit = 100;
relax = 1.0;
x0 = 0.2;
x1 = 0.3;

alpha = q3_p3_Secant(@q3_p3_NLAE,x0,x1,tol,maxit,relax)
hold on
fplot(@q3_p3_NLAE,[0.01,0.99],'-g')
fplot(@(alpha)(0),[0.01,0.99],'-r')
xlabel('alpha [-]')
ylabel('f(alpha) = m_{12}(alpha) - 1.0   [kg/hour]')
legend('function','zero reference')

%{
Abbreviated output:

F_COUNT                 Xk              F(Xk)       Xk_tol              F_tol
      1         2.0000e-01         1.7908e+00
      2         3.0000e-01         1.5533e+00
      3         9.5399e-01        -7.6085e-01   6.8553e-01          7.6085e-01
      4         7.3897e-01         1.9038e-01   2.9097e-01          1.9038e-01
      5         7.8200e-01         1.9134e-02   5.5032e-02          1.9134e-02
      6         7.8681e-01        -5.3646e-04   6.1112e-03          5.3646e-04

alpha =
    0.7868
%}
```

Python code

```
import numpy as np
from numpy.linalg import *
from matplotlib.pylab import *
```

```python
def secant_method(func, x0, alpha=1.0, tol=1E-9, maxit=200):
    """
    Uses the secant method to find f(x)=0.
    INPUTS
        * f     : function f(x)
        * x0    : initial guess for x
        * alpha : relaxation coefficient: modifies Secant step size
        * tol   : convergence tolerance
        * maxit : maximum number of iteration, default=200
    """
    # You can write more than one command per line in Python:
    x, xprev = x0, 1.1*x0
    f, fprev = func(x), func(xprev)
    rel_step = 2.0 *tol
    k = 0
    print('{0:12} {1:12} {2:12} {3:12} {4:12}'\
    .format('Iteration', 'x', 'f(x)', 'Rel step', 'alpha * Delta x'))

    while (abs(f) > tol) and (rel_step) > tol and (k<maxit):
        rel_step = abs(x-xprev)/abs(x)

        # Full secant step
        dx = -f/(f - fprev)*(x - xprev)

        # Update 'xprev' and 'x' simultaneously
        xprev, x = x, x + alpha*dx

        # Update 'fprev' and 'f':
        fprev, f = f, func(x)

        k += 1

        print('{0:10d} {1:12.5f} {2:12.5f} {3:12.5f} {4:12.5g}'\
               .format(k, xprev, fprev, rel_step, alpha*dx))

    return xprev

def matrix_A(alpha, XA, XD, XE):
    """
    Calculates and returns the A matrix that represents the reactor network.

    A has 12 rows (equations) and 12 columns (unknowns m1, m2 ... m12) for
    the given equations.
    """
    a = alpha
    A = np.array(
    [[1,       0,  0,  0,      0,      0,      0,         0,  0,       0,  0,  0],
     [0,       1,  0,  0,      0,      0,      0,         0,  0,       0,  0,  0],
     [-(1-XA), 0,  1,  0,      0,      0,      0,         0,  0, -(1-XA),  0,  0],
     [0,      -1, -1,  1,      0,      0,      0,         0, -1,       0,  0,  0],
     [0,       0,  0, -1,      1,      1,      0,         0,  0,       1, -1,  0],
     [0,       0,  0,  0,     -1,      1,      0,         0,  0,       0,  0,  0],
     [0,       0,  0,  0,-(1-XD),      0,      1,         0,  1,       0,  0,  0],
     [0,       0,  0,  0,      0,-(1-XE),-(1-XE),         1,  0,       0,  0,  0],
     [0,       0,  0,  0,      0,      0,     -1,         0,  1,       0,  0,  0],
     [0,       0,  0,  0,      0,     -1,      0,         0,  0,       1,  0,  0],
     [0,       0,  0,  0,      0,      0,      0,    -alpha,  0,       0,  1,  0],
     [0,       0,  0,  0,      0,      0,      0, -(1-alpha),  0,       0,  0,  1],
    ])
    return A
```

```python
alpha = 0.20;
XA = 0.40;
XD = 0.82;
XE = 0.15;

def func(x):
    A = matrix_A(x, XA, XD, XE)
    x = solve(A, b)
    return x[11, 0] - 1.0   # f(x) = m_12 - 1.0


b = np.zeros((12,1))
b[0], b[1] = 10.0, 2.0

alpha_start = 0.2
A = matrix_A(alpha_start, XA, XD, XE)
print(np.round(A,2))
x = solve(A, b)
print(np.round(x.T, 2))

alpha_desired= secant_method(func, x0=alpha_start, tol=1E-3)
print(alpha_desired)

# For plotting figure
alpha_range = np.arange(0, 1.01, 0.01)
m12 = np.zeros(alpha_range.shape)
for k, alpha in enumerate(alpha_range):
    A = matrix_A(alpha, XA, XD, XE)
    x = solve(A, b)
    m12[k] = x[11]
fig = figure(figsize=(9,4))
plot(alpha_range, m12, '-')
axhline(y=1.0, color='r')
grid('on')
xlabel('alpha')
ylabel('$m_{12}$')
fig.savefig('../images/take-home-midterm-Q3-alpha.png')
```

Question 4 [6 = 2 + 4]

Note: Even though this question is typical of material learnt about in later courses, you are still able to solve this question using tools just covered in this course.

A polymer compound is being held in a small stirred tank that is externally heated. The polymer leaving the tank must have a certain viscosity so that the downstream processes can operate as required. The primary method of adjusting the viscosity is by setting the temperature of the tank. Several experiments, performed by your colleagues, have determined the system behaves in a second-order manner (you will learn how to determine this in your process control course).

$$\tau^2 \frac{\mathrm{d}^2\mu(t)}{\mathrm{d}t^2} + \tau \frac{\mathrm{d}\mu(t)}{\mathrm{d}t} = K_p(T(t) - 318)$$

where:

- $\mu(t)$ is the viscosity measured in centipoise, cP and the target viscosity required is 30 cP
- $T(t)$ is the tank temperature, measured in Kelvin
- $K_p = -0.25$ cP/K
- $\tau = 2.1$ minutes

A feedback control system is implemented to keep μ at its target of 30 cP. The type of feedback controller used is a *proportional-only* controller, meaning the tank temperature, $T(t)$ is adjusted in proportion to the deviation (error) in viscosity, as given by:

$$T(t) = K_c\big[30 - \mu(t)\big] + 318$$

where $K_c = -1.9$ K/cP. At steady state, the tank temperature is normally set to 318 K to maintain the required viscosity of 30 cP.

1. According to this feedback control law, what should the tank temperature be set to if the viscosity is currently at 35 cP? And for 28 cP? Comment whether or not these answers make sense.
2. Someone has turned the tank's heating system off overnight and the viscosity of the material is currently at 40 cP. Create a plot, as a function of time, that shows how the material's viscosity is brought back to the target of 30 cP. Let $t = 0$ be the moment the heating system is turned back on. Using your plot, estimate how much time will be required to reach **stable** operation (i.e. return to a constant viscosity of 30 cP)?

Bonus question [1 mark]: The value K_c is called the feedback controller's gain (aggressiveness). Show your plot from part 2 of the question, but superimpose the time-varying viscosity curves for two other values of K_c, one higher and one lower. How would you adjust K_c to achieve the target of 30 cP in a shorter amount of time? Can you suggest a better K_c value than the current setting?

Solution

1. The question tells us that at steady state, the temperature is set to 318K to maintain the viscosity at 30cP. We can see this also by substituting into $T(t) = K_c\big[30 - \mu(t)\big] + 318 = -1.9\big[30 - 30\big] + 318 = 318$ K. But when the viscosity starts to deviate from target:

   - e.g. at 35 cP, then the tank temperature is set to $T(t) = -1.9\big[30 - 35\big] + 318$ = 327.5 K
   - e.g. at 28 cP, then $T(t) = -1.9\big[30 - 28\big] + 318 = 314.2$ K.

   This values do makes sense: when $\mu = 35$ cP (viscosity is above target) the temperature is increased above 318K, meaning that the viscosity will start to decrease to reach the target of 30 cP. As the viscosity gets closer and closer to the target, the temperature gets closer to its steady-state value of 318 K.

   Similarly if the viscosity is too low, then the temperature is lowered so that the viscosity will rise with time.

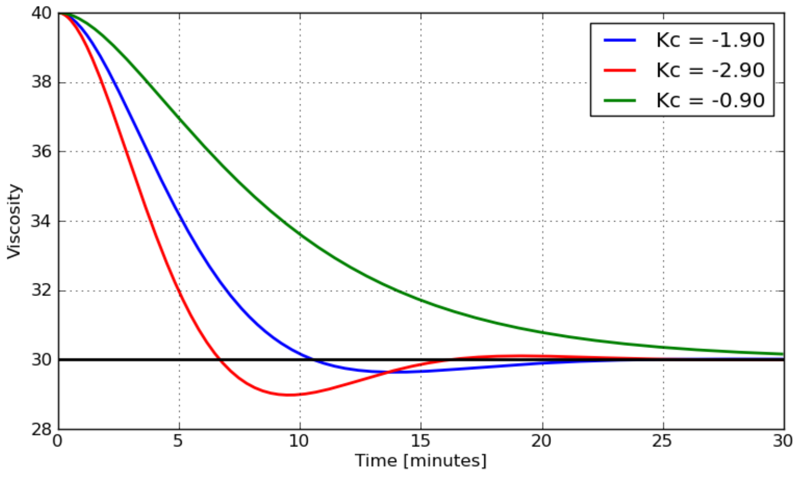
2. *Grading*: 1 to convert to coupled ODEs, 1 for initial conditions, 1 for plot, 1 for determining stable operation (any reasonable answer got full grade).

   We must convert the second order ODE system to a set of two coupled first-order ODEs. Let $y_1 = \mu$ and let $y_2 = \frac{d\mu}{dt}$. Then:

$$\frac{dy_1}{dt} = \frac{d\mu}{dt} = y_2$$
$$\frac{dy_2}{dt} = \frac{K_p(T(t) - 318.0)}{\tau^2} - y_2\tau$$

   with the algebraic equation $T(t) = K_c\big[30 - \mu(t)\big] + 318$. Please note: there is no need to substitute these values into the derivatives and simplify. Rather let the computer do the calculations, than you making an error.

   The plot from the ODE integrator output (code is shown also) shows the system reaching a stable point at about 24 minutes.



   The Python code used to generate the above figure is shown here. MATLAB code would be very similar.

```
import numpy as np
from scipy import integrate
from matplotlib.pylab import *

def dmu_dt(t, y, Kc):
    """ The derivative functions

    Definitions
    -----------
    y1 = mu(t)          and initial conditions are: mu(t=0)  = given
    y2 = d(y1)/dt                                   d(y1)/dt = assumed to be zero
    """
    # Constants
    tau = 2.1           # minutes
    Kp  = -0.25         # cp/K

    # Assign some variables for convenience of notation
    mu      = y[0]
    dmu_dt  = y[1]

    # Algebraic equations
    T = Kc * (30 - mu) + 318.0

    # Output from ODE function must be a COLUMN vector, with n rows
    n = len(y)      # 2: implies we have two ODEs
    dydt = np.zeros((n,1))
    dydt[0] = dmu_dt
    dydt[1] =  (Kp * (T-318.0))/tau**2 - dmu_dt/tau
    return dydt

def integrate_and_plot(Kc, colour):
    """ Integrate the ODE system with the given value of ``Kc``;
        plot the viscosity trajectory with the given ``colour``.
    """

    # Use a proper integrator
    r = integrate.ode(dmu_dt)
    r.set_integrator('vode', method='bdf')
    r.set_f_params(Kc)

    # Set the time range
    t_start = 0.0
    t_final = 30.0
    delta_t = 0.11
    # Number of time steps: 1 extra for initial condition
    num_steps = np.floor((t_final - t_start)/delta_t) + 1

    # Set initial condition(s): for integrating variable and time!
    mu_t_zero = 40.0      # cP
    dmu_dt_t_zero = 0.0   # cP/min: assume first derivative is constant
    r.set_initial_value([mu_t_zero, dmu_dt_t_zero], t_start)

    # Additional Python step: create vectors to store trajectories
    t = np.zeros((num_steps, 1))
    mu = np.zeros((num_steps, 1))
    t[0] = t_start
    mu[0] = mu_t_zero
    # We aren't really interesting in the 1st derivative

    # Integrate the ODE(s) across each delta_t timestep
    k = 1
    while r.successful() and k < num_steps:
        r.integrate(r.t + delta_t)

        # Store the results to plot later
        t[k] = r.t
        mu[k] = r.y[0]
        k += 1

    plot(t, mu, colour, linewidth=2, label='Kc = %1.2f' % Kc)

if __name__ == '__main__':

    fig = figure(figsize=(9,5))
    integrate_and_plot(Kc = -1.9, colour='b')
    integrate_and_plot(Kc = -2.9, colour='r')
    integrate_and_plot(Kc = -0.9, colour='g')
    axhline(y=30.0, color='k', linewidth=2)
    grid('on')
    xlabel('Time [minutes]')
    ylabel('Viscosity')
    legend()
    fig.savefig('../images/take-home-mideterm-feedback-control.png')
```

Question 5 [5]

This question considers a pilot-plant CSTR, liquid reaction process, fed with reactants A, B and C. There are two reactions taking place in the system:

$$\mathsf{A} + \mathsf{B} \rightarrow 2\mathsf{D} + \mathsf{E}$$
$$2\mathsf{C} + \mathsf{D} \rightarrow \mathsf{F} + \mathsf{E}$$

The desired products are E and F, while species D is an intermediate. From economic considerations, it is desirable to operate the reactor to maximize the production of E and F. This is achieved by minimizing the amount of unreacted D in the outlet from the reactor.

One way this can be done is by adjusting the tank's jacket temperature, which in turn affects the reaction kinetics and alters the balance of products in the outlet. Unfortunately the first principles rate expressions for the reactions are not known, so derivation of mass and energy balances to find this jacket temperature is not possible.

You have time and resources to run the pilot-plant reactor at only 8 different jacket temperatures, between a minimum and maximum range of 278 K to 355 K. You are expected to use any of the tools from this course to find a jacket temperature that leads to the lowest possible concentration of D [mol/L]. You may use of the fact (learnt in your calculus course) that the optimum – either a maximum or a minimum – of a function $f(x)$ occurs at $\dfrac{df(x)}{dx} = 0$; and that optimum is a minimum if $\dfrac{df^2(x)}{dx^2} > 0$.

A model of the process has been simulated, and is available on the course website. Nominate *one of your group members* and email their name and student number to the course instructor, together with the name of the other, optional, group member. This will activate an account for your group. Once you sign into the account you will be able to enter in various values of the jacket temperature, and the server will simulate the experiment and return the corresponding concentration of D, as if the pilot plant was operated at that jacket temperature.

Four of the grading points for this question will be marked mainly on the *description of your strategy and the systematic methodology followed* to locate a suitable jacket temperature. Make sure you explain your approach clearly and present good visualization plot(s), exactly as you would have to in a report to your manager. Your final answer (one point) must provide a recommendation to your manager for the jacket temperature.

Also note:

- There is error (noise) in the concentration measurements of ± 1.0 mol/L.
- The server will return different results for each group.
- Please enter your jacket temperature settings carefully - if you input incorrect values you will have to work with those results. **You will only be able to run at most 8 experiments.**
- The server will also keep track of and display all your previous experiments on a results sheet.

Once you have completed the question, include the result sheet from the server in your answerr. (The true jacket temperature for your team will be be available on the website after the exam is handed in). Please note, there is more than one correct method to approach the problem.

There is **1 bonus mark** if you are able to locate the jacket temperature within ± 0.5 K and additional bonus marks as follows: $\dfrac{\max\{6 - N, 0\}}{4}$, where N is the number of experiments you used.

Solution

Grading was awarded as follows (with discretion for cases that didn't quite meet all these criteria):

- [1] for discussing how the initial points were chosen (equally spaced, or some other systematic, i.e. non-random, way): though using **all your experiments** with equally spaced points makes little sense, so 0.25 was deducted for that.

- [1] discussion of any systematic method used (e.g. Newton polynomials, spline, *etc*)
- [1] for discussing how error in the concentration values will affect your method chosen (spline and polynomial fitting assumes no error, but you can accommodate for it by fitting the points relative far apart, so error doesn't affect your results too much)
- [1] for good, clear plots that show intermediate curve fitting, and how/why you decided to add extra experiments, or stop adding experiments
- [1] for your calculations from the curve fit to present your final answer; note that the final answer need not be an experiment that was actually run
- Bonus grading was changed so that you got 1 mark if you were within ± 0.5 K of the true value (original it was ± 0.25 K, which was quite a demanding criterion).

A 0.75 mark was deducted if you:

- fit splines/polynomials through points close together, since there is error and you will get thrown off
- fit a spline over the entire range of data, which is also not suitable since you cannot hope to capture the phenomena from only a few data points over a wide range

A 0.25 mark was deducted if plots weren't clear, experimental points weren't labelled (managers find it easier to understand your approach on a clear plot than reading paragraphs of text).

Some notes regarding this question:

- When fitting a function to points, equally spaced points are a good starting point if you have no other prior knowledge. However, you will get better information if you used four points at roughly 15%, 35%, 65%, 85% across the range. Most groups fit 5 points equally spaced. Points right at the lower and upper bound can be avoided: I would rather fit 4 than 5 points - if you need to fit a point at 0% or 100% then you have the option later on.
- However, as you start to accumulate points you now have some inference of the system's behaviour and you should adjust your strategy.
- For example, if you see your minimum in the left side of the plot, do not add any points to the right side: these are not going to improve your fit.
- Can view this question as solving $f'(x) = 0$, but you have to first find $f(x)$ by selecting points and fitting some sort of curve.
- Don't place points close to each other, to account for error in D. Also not too far from each, so that you model the local minimum.
- Point placement is not an easy problem, as most of you resorted to evenly spaced points. The best time to do an experiment is after the experiment.
- Place your points where you will get maximum information (value) from them.
- Don't fit the spline across points from beginning to end of the range.

I showed examples in class of various cases that helped illustrate these points.

END